



Universidad
Andrés Bello

Solución Actividad 1 Evaluada S_554 y Ejemplo Matriz

Estructuras de Datos

Docente: Pamela Landero Sepúlveda
p.landero@uandresbello.edu

Formar

Transformar



Universidad
Andrés Bello

ENUNCIADO ACTIVIDAD 1 EVALUADA

Forme grupos de 2 o 3 estudiantes e implemente en C lo siguiente:

- 1) Desarrolle una función que reciba un entero n y devuelva un arreglo de largo n con los primeros n números de Fibonacci. Implemente además el main asociado.

$$f_n = \begin{cases} 0 & \text{si } n = 0 \\ 1 & \text{si } n = 1 \\ f_{n-1} + f_{n-2} & \text{si } n > 1 \end{cases}$$

- 2) Desarrolle una función que reciba un arreglo dinámico y recorra sus elementos verificando la paridad de sus elementos. Implemente además el main asociado.

ACTIVIDAD 1 - EJERCICIO 1

Desarrolle una función que reciba un entero n y devuelva un arreglo de largo n con los Primeros n números de Fibonacci

VERSION 1: SIN FUNCIONES

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]) {
    int n, i;
    // LEER n
    printf("Ingrese valor de n:");
    scanf("%d",&n);
    if (n<=0){
        printf("\n Valor de n es  %d  no es posible crear arreglo",n);
    }else{
        // INSERTA VALORES FIBONACCI AL ARREGLO
        int fib[n]; //TAMAÑO FIJO
        for (i=0;i<n;i++){
            if (i==0){
                fib[0]=0;
            }else{
                if(i==1){
                    fib[1]=1;
                }else{
                    fib[i]=fib[i-2]+fib[i-1];
                }
            }
        }

        for (i=0;i<n;i++){
            printf("\n i=%d tiene valor:%d ",i,fib[i]);
            printf( "   Es lo mismo que: *(fib + %d) : %d\n", i, *(fib + i));
        }
    }
    return 0;
}
```

ACTIVIDAD 1 - EJERCICIO 1

VERSION 2: CON FUNCIONES

FUNCIÓN CREA EL ARREGLO, LE INSERTA DATOS Y RETORNA EL ARREGLO

```
int main(int argc, char *argv[]) {
    int n, i;
    n = leer_n(); //llama la función y recibe el valor de n en OTRA VARIABLE n
    if (n<=0){
        printf("\n Valor de n es  %d  no es posible crear arreglo",n);
    }else{
        // LAS FUNCIONES QUE RETORNAN UN ARREGLO, RETORNAN UN PUNTERO (dirección).
        // CREAR UN PUNTERO QUE RECIBA EL VECTOR CREADO POR LA FUNCIÓN
        // fib toma el valor de la dirección que posee obtenerFib
        int *fib = obtenerFib(n);
        //
        for (i=0;i<n;i++){
            printf("\n i=%d tiene valor:%d ",i,fib[i]);
            printf( "   Es lo mismo que: *(fib + %d) : %d\n", i, *(fib + i));
        }
    }
    return 0;
}

//Leer valor de n y retornar el valor al main
int leer_n(void){
    int n;
    printf("Ingrese valor de n:");
    scanf("%d",&n);
    return n;
}
```

```
//CREAR EL ARREGLO E INSERTARLE VALORES PARA LUEGO RETORNARLO AL MAIN
//esta función es de tipo puntero porque retorna una dirección
int *obtenerFib(int n){
    int *arrayFib = (int *) malloc( n*sizeof(int));
    int i;
    for (i=0;i<n;i++){
        if (i==0){
            arrayFib[0]=0;
        }else{
            if(i==1){
                arrayFib[1]=1;
            }else{
                arrayFib[i]=arrayFib[i-2]+arrayFib[i-1];
            }
        }
    }
    return arrayFib; // retorna la dirección de la primera posición del arreglo
}
```

ACTIVIDAD 1 - EJERCICIO 1: FIBONACCI

VERSION 3: CON FUNCIONES

FUNCIÓN QUE INSERTA DATOS AL ARREGLO CREADO EN EL MAIN. RETORNA VOID

```
int main(int argc, char *argv[]) {
    int n, i;
    n = leer_n(); //llama la función y recibe su valor de retorno en n
    if (n<=0){
        printf("\n Valor de n es  %d  no es posible crear arreglo",n);
    }else{
        // SE CREA EL ARREGLO AQUI EN EL MAIN
        int *fib = (int *) malloc( n*sizeof(int));
        // SE PASA EL ARREGLO POR REFERENCIA A LA FUNCIÓN. DE ESTA FORMA,
        // LOS VALORES INGRESADOS AL VECTOR EN LA FUNCIÓN SE PUEDEN VER
        // DESDE EL MAIN SIN QUE LA FUNCIÓN RETORNE EL VECTOR.
        obtenerFib(n, fib); //función void (no se asigna a una variable)
        //
        for (i=0;i<n;i++){
            printf("\n i=%d tiene valor:%d ",i,fib[i]);
            printf( "   Es lo mismo que: *(fib + %d) : %d\n", i, *(fib + i));
        }
        free(fib);
    }
    return 0;
}

//leer valor de n y retornarlo al main
int leer_n(void){
    int n;
    printf("Ingrese valor de n:");
    scanf("%d",&n);
    return n;
}
```

```
//ACTUALIZA EL ARREGLO CREADO EN EL MAIN CON LOS VALORES FIBONACCI
void obtenerFib(int n, int *pfib){
    //El arreglo se crea en el main, por lo tanto,
    //se recibe por parámetro (por referencia) para actualizarlo
    int i;
    for (i=0;i<n;i++){
        if (i==0){
            pfib[0]=0;
        }else{
            if(i==1){
                pfib[1]=1;
            }else{
                pfib[i]=pfib[i-2]+pfib[i-1];
            }
        }
    }
}
```

ACTIVIDAD 1 - EJERCICIO 2

Desarrolle una función que reciba un arreglo dinámico y recorra sus elementos verificando la paridad de dichos elementos

VERSION CON FUNCIONES

```
int main(int argc, char *argv[]) {
    int n, i;
    n = leer_n(); //llama la función y recibe el valor de retorno en n
    if (n<=0){
        printf("\n Valor de n es  %d  no es posible crear arreglo",n);
    }else{
        //SE CREA UN ARREGLO DINÁMICO
        int *arr = (int *) malloc( n*sizeof(int));
        // SE PASA EL ARREGLO POR REFERENCIA A LA FUNCIÓN. DE ESTA FORMA,
        // LOS VALORES INGRESADOS AL VECTOR EN LA FUNCIÓN SE PUEDEN VER
        // DESDE EL MAIN SIN QUE LA FUNCIÓN RETORNE EL VECTOR.
        insertarValores(n,arr);
        revisarParidad(n, arr); //función void (no se asigna a una variable)
        //
    }
    return 0;
}
```

```
//Leer cantidad de elementos y retornar el valor al main
int leer_n(void){
    int n;
    printf("Ingrese cantidad de elementos del arreglo:");
    scanf("%d",&n);
    return n;
}
```

```
//INGRESAR VALORES AL ARREGLO. SE PUEDEN LEER POR TECLADO.
//EN ESTA OPORTUNIDAD SE ASIGNARÁN VALORES ALEATORIOS
void insertarValores(int n, int *parr){ //esta función es de tipo void
    //El arreglo se crea en el main y se recibe por parámetro (por referencia)
    int i;
    printf("\nVALORES DEL ARREGLO");
    for (i=0;i<n;i++){
        parr[i]= rand (); //valores entre 0 y un valor muy grande
        //parr[i]= rand () % 11; //valores entre 0 y 10
        //parr[i]= rand () % 11 + 10; //valores entre 10 y 20
        //parr[i]= rand () % 11 + 20; //valores entre 20 y 30
        printf("\n Valor %d:  %d ",i,parr[i]);
    }
}
```

```
//REVISAR EL ARREGLO QUE ES DEL MAIN E INDICA CUÁLES SON PARES
void revisarParidad(int n, int *parr){ //esta función es de tipo void
    //El arreglo se crea en el main, por lo tanto se recibe por parámetro (por referencia)
    int i;
    for (i=0;i<n;i++){
        if (parr[i]%2==0){
            printf("\n Valor %d  es par: %d ",i,parr[i]);
        }
    }
}
```

EJEMPLO DE MATRICES DINÁMICAS

Ingresar todas las notas parciales de los alumnos

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]){
    float **mtr_notasAlum; //matriz
    int i , j, filas , columnas;
    printf ("Ingrese número de Alumnos (Filas) : ") ;
    scanf( " %d" , &filas ) ;
    printf ("Ingrese número de Notas por Alumno (Columnas) : ") ;
    scanf( " %d" , &columnas ) ;

    /* reserva de memoria para las filas*/ //Cada fila representará a un alumno
    mtr_notasAlum = (float **) malloc (filas*sizeof(float *));

    for ( i =0; i<filas; i++) {
        //para cada fila reserva memoria para las columnas
        mtr_notasAlum[i] = (float *) malloc (columnas*sizeof(float));
    }
    /* Se ingresan las notas de cada alumno */
    for ( i =0; i<filas ; i++) { //alumnos
        for ( j =0; j<columnas; j++) { //notas de cada alumno
            printf("Ingrese nota %d del alumno %d: ",j,i);
            scanf("%f", &mtr_notasAlum[i][j]);
        }
        printf("\n");
    }
}
```

//TAREA:

- // 1) LLAMAR A UNA FUNCIÓN QUE CALCULE EL PROMEDIO POR ALUMNO (GUARDAR EN UN ARREGLO)
- // 2) LLAMAR A UNA FUNCIÓN QUE MUESTRE LOS ALUMNOS REPROBADOS (PROMEDIO MENOR QUE 4)
- // 3) MOSTRAR EL PROMEDIO GENERAL DEL CURSO

```
//MOSTRAR NOTAS
for ( i =0; i<filas ; i++) { //alumnos
    printf("NOTAS ALUMNO %d: ",i);
    for ( j =0; j<columnas; j++) { //notas de cada alumno
        printf(" %2.1f\t", mtr_notasAlum[i][j]); /*mostrar la nota*/
    }
    printf("\n");
}

/* LIBERACIÓN MEMORIA */
for ( i =0; i<filas ; i++) {
    free (mtr_notasAlum[ i ] ) ;
}
free (mtr_notasAlum) ;
return 0;
```